High-Speed Networking in Clusters without OS-bypass and Zero-copy

Brice Goglin

High-Speed Networking in Clusters without OS-bypass and Zero-copy
(Short Paper)

Brice Goglin
Myricom, Inc. – Arcadia, CA 91006 – USA
Laboratoire de l'Informatique du Parallélisme – CNRS-ENS Lyon-INRIA-UCBL – France
Brice.Goglin@ens-lyon.org

Abstract

High-performance computing requires low latency and high bandwidth communications. The emergence of high-speed networks ten years ago has lead to the development of advanced software strategies to provide high performance, including OS-bypass and zero-copy. We show that their impact is not so important nowadays and that almost the same performance might be achievable by using traditional models which are easier to implement.

1 Introduction

High-performance computing requires communications to be as fast as possible. Since the emergence of parallel applications, massive supercomputers have been replaced by clusters of workstations which are more generic and extensible, and less expensive. As the application always require more computational power, high-speed local networks with intelligent interface cards have been developed, such as MYRINET or INFINIBAND.

Very low latency (a few microseconds) and very high bandwidth (hundreds or thousands of megabytes per second) may now be achieved using custom software layers that have been designed and optimized for user-level communications between nodes running a parallel application. Such performance has been reached by using innovative software and hardware techniques which include zero-copy communications and bypassing the operating system (*OS-bypass*).

These strategies place strong requirements on the support software because of the need to avoid the expense of using system calls and intermediate copies. However, processor and network technologies evolved in many ways since their design. It is thus important to recheck the assumptions that led to these strategies. The relative cost of memory copy and system calls seems to be lower now. Hence, we are proposing here to study the impact of zero-copy and OS-bypass on today's hardware.

2 Zero-copy and OS-bypass models

Achieving low latency requires to reduce the critical path from the application to the network. Thus, system calls are avoided by posting communication requests from the application. Meanwhile, zero-copy enables communication without wasting CPU cycles that computing applications need. But, zero-copy OS-bypass implementation raises the problem of translating virtual addresses that the application manipulates into physical addresses that the network interface uses for DMA.

This problem has been addressed by pining pages in physical memory and storing their address translation in the NIC [6]. However, this strategy is often expensive and has thus been optimized by delaying unpining with a pin-down cache [5]. Even if it is efficient, this model has raised multiple strong technical issues, including the need to intercept memory management calls from the application and replacing the memory allocator. Moreover, we showed that it is very difficult for communication initiated in the LINUX kernel because of a lack of software support for high-speed networks [3].

3 Using Memory Copies

Zero-copy is known to be useless for small messages since a memory copy is faster than memory pining [4]. As memory performance has increased, zero-copy is actually less often used now than previously. For instance the *Myrinet Express* driver (MX) uses copy up-to 32 kilobytes messages.

We studied in [2] the removal of this copy when the physical addresses of target pages are known and contiguous. Figure 1 presents the results and shows that the memory copy reduces the bandwidth that the application observes. But, an efficient pipeline of the copy and network transmission reduces the difference. This result proves that zero-copy is not so efficient for limited size messages and even almost useless for less 4 kB.

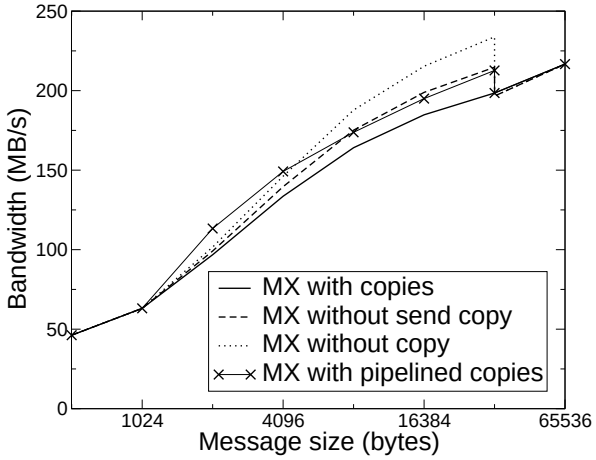

Figure 1. Impact of the removal of the copy for medium messages in Myrinet Express.

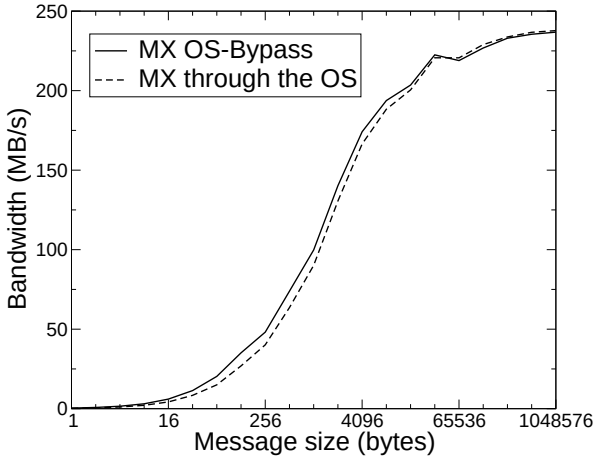

Figure 2. Impact of the removal of OS-bypass on the Myrinet Express performance.

However, memory copies are known to waste CPU cycles and pollute the cache. We are currently studying *non-temporal* copy instructions which are supposed to avoid any pollution. In the meantime, INTEL is planning to add a DMA engine in the next processor generation in order to offload memory copies [1]. This kind of feature, which is similar to multicores that are specialized for memory copies, enables memory copy in the background while the application continues computing. One drawback is the requirement to pass physical addresses to this DMA engine, which requires the operating system intervention to translate virtual addresses. This is another reason why we now look at removing the OS-bypass strategy.

4 Removal of OS-bypass

OS-bypass has been designed to reduce the communication latency up-to less than 2 microseconds on QSNET or INFINIPATH. Modern machines may, however, process a system call very quickly. We measured 75 ns on a 2.2 GHz OPTERON. This leads us to study non-OS-bypass high-speed networking.

As a proof of concept, we implemented a non-OS-bypass MYRINET MX by replacing each application call with a syscall using kernel library communications. Figure 2 shows that we achieve almost the same bandwidth for large messages while the latency is 3.3 μs instead of 2.4. This prototype may still need large optimisations. We already observed a 300 ns latency improvement by avoid multiple system calls during busy loop when waiting for completion and factorizing locks. Even if the latency is 20 % higher, the simple implementation shows that OS-bypass may not be so required for high-speed networking.

Keeping the operating system in the critical path presents two advantages. First, accessing the hardware requires access rights checks. Since the user library may not be trusted, OS-bypass leads to checking in the NIC, while its cycles are expensive and slow. Checking in the kernel would be more efficient and easy. We expect a 0.1 μs latency reduction in MX by removing those checks.

Secondly, it is sometimes interesting to share a communication channel between processes, especially because the number of channels is limited by the hardware resources while the number of processors keeps growing. Sharing in user-space is not easily possible while the kernel intervention makes it trivial. These two reasons may justify keeping the OS in the critical path even if the latency is slightly higher.

5 Conclusion and future works

We propose in this paper to study the validity of OS-bypass and zero-copy on recent hardware. Our prototype implementation of MYRINET MX enables easier software support with almost similar performance. Zero-copy does only achieve better performance for large messages while small message latency is only 20 % higher when the operating system is involved. Moreover, the operating system makes it very easier to check access rights and share communication channels.

We are now working on reducing the impact of memory copies on the application performance. Non-temporal memory access enables memory copy without polluting the cache, while copy offload in next generation processors should drastically reduce the impact. Such an implementation may imply a large redesign of the software layers supporting high-networks and may benefit from operating system support if OS-bypass is not involved anymore.

References

[1] Accelerating High-Speed Networking with Intel I/O Acceleration Technology, May 2005. `http://www.intel.com/technology/ioacceleration/306517.pdf`.

[2] B. Goglin, O. Glück, and P. V.-B. Primet. An Efficient Network API for in-Kernel Applications in Clusters. In *Proceedings of the IEEE International Conference on Cluster Computing*, Boston, Massachussets, Sept. 2005. IEEE Computer Society Press.

[3] B. Goglin, L. Prylli, and O. Glück. Optimizations of Client's side communications in a Distributed File System within a Myrinet Cluster. In *Proceedings of the IEEE Workshop on High-Speed Local Networks (HSLN), held in conjunction with the 29th IEEE LCN Conference*, pages 726–733, Tampa, Florida, Nov. 2004. IEEE Computer Society Press.

[4] L. Prylli and B. Tourancheau. Protocol Design for High Performance Networking: a Myrinet Experience. Technical Report 97-22, LIP-ENS Lyon, 69364 Lyon, France, 1997.

[5] H. Tezuka, F. O'Carroll, A. Hori, and Y. Ishikawa. Pin-down cache: A Virtual Memory Management Technique for Zero-copy Communication. In *12th International Parallel Processing Symposium*, pages 308–315, Apr. 1998.

[6] M. Welsh, A. Basu, and T. von Eicken. Incorporating Memory Management into User-Level Network Interfaces. In *Proceedings of Hot Interconnects V*, Stanford, Aug. 1997.